REMARKS

Claims 1-44 have been rejected under 35 U.S.C. §112, first paragraph, as failing to comply with the written description requirement.

Claims 1-44 have been rejected under 35 U.S.C. §112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter, which applicant regards as the invention.

Claims 1, 4, 6, 8-11, 13, 14, 16, 18-21, 31, 32, 37, and 39-42 have been rejected under 35 U.S.C. §103(a) as being unpatentable over "Description of the Prior Art" ("DPA") in view of U.S. Patent No. 6,023,704 to Gerard et al. ("Gerard"), and further in view of "Operating System Concepts" by Silberschatz and Galvin ("Silberschatz").

Claims 2, 3, 28-30, 33, 34, and 44 have been rejected under 35 U.S.C. §103(a) as being unpatentable over DPA, Gerard, and Silberschatz, and further in view of "Linkers and Loaders" by Levine ("Levine").

Claims 5, 15, 23, and 36 have been rejected under 35 U.S.C. §103(a) as being unpatentable over DPA, Gerard, and Silberschatz, and further in view of "Understanding Computer: Input/Output" by Time-Life Books ("Time-Life").

Claims 7, 17, and 38 have been rejected under 35 U.S.C. §103(a) as being unpatentable over DPA, Gerard, Silberschatz, and Levine, and further in view of U.S. Patent No. 4,974,191 to Amirghodsi et al. ("Amirghodsi").

Claim 24 has been rejected under 35 U.S.C. §103(a) as being unpatentable over DPA, Gerard, Silberschatz, and Time-Life, and further in view of "Microsoft Computer Dictionary" published by Microsoft Press ("MCD").

Claims 12 and 43 have been rejected under 35 U.S.C. §103(a) as being unpatentable over DPA, Gerard, and Silberschatz and further in view of U.S. Patent No. 6,237,091 to Firooz et al. ("Firooz").

Claims 22 and 25-27 have been rejected under 35 U.S.C. §103(a) as being unpatentable over DPA, Gerard, Silberschatz, and Firooz, and further in view of U.S. Patent No. 5,132,716 to Samuels et al. ("Samuels").

Claims 3, 29, and 34 have been canceled.

Claims 1-2, 4-28, 30-33, and 35-44 remain pending.

Rejection of Claims 1-44 under 35 U.S.C. §112, first paragraph

The Office Action state that newly amended independent claims 1, 13, 31, 32, and 44 each contain limitations reciting "...and thereby prevent a task switching function that originates from the one or more code update routines of the incoming image from executing" and that these limitations are not supported by the originally-filed specification.

Claims 1, 13, 31, 32, and 44 have been amended to comply with the written description requirement. Full support for the claim amendments can found on page 14, lines 10-16 of the specification and respectfully, no new matter is being entered.

Applicant submits that this rejection is now overcome.

Rejection of Claims 1-44 under 35 U.S.C. §112, second paragraph

The Office Action states that the word "thereby" in the newly amended independent claims 1, 13, 31, 32, and 44 appears to make the associated steps optional and that it is not clear if the use of the word and the associated limitations would limit the scope of the claims.

Claims 1, 13, 31, 32, and 44 have been amended to particularly point out and distinctly claim the subject matter, which applicant regards as the invention. Specifically, the word "thereby" and phrase associated therewith has been deleted.

Applicant submits that this rejection is now overcome.

Rejection of Claims 1, 4, 6, 8-11, 13, 14, 16, 18-21, 31, 32, 37, and 39-42 under 35 U.S.C. §103(a)

The Office Action states that the DPA teaches the steps of independent claim 1 where step (a) is disclosed by the text "Thus, the system firmware on the PROM may be updated because the current system firmware is executed from the RAM; step (b) is disclosed by the text "Utilizing code update routines from the firmware update itself has associated advantages"; and step (c) is disclosed by the text "...a code update to the firmware may occur while the system operates normally utilizing one or more other process threads of the firmware, thereby accomplishing a background code update to the firmware." The Office Action further states that in the context of normally utilizing threads with a background code update, a process of normally switching tasks in order to provide the background update is described. Since task switching is normally handled by the current code image, normal utilization of threads in terms of task switching is handled by the current code image.

It is again respectfully submitted that the cited text from the DPA has been taken out of context and mischaracterized. Cited references must be in interpreted in light of the totality of what they disclose. Specifically, the text "Therefore, by executing firmware from a separate memory device, a code update to the firmware may occur while the system operates normally utilizing one or more other process threads of the firmware, thereby accomplishing a background code update to the firmware" refers to executing a current code image from a separate memory device, such as a

RAM, or PROM, for example, while an incoming code image is received and stored on the only PROM (in the case where the current code image is executing from RAM) or stored on a second PROM (in the case were the current code image is executing from a first PROM). While the incoming code image is being stored, the current code image is free to utilize a process thread or switch between multiple process threads. However, as stated in the DPA (see page 4, lines 14 to page 5, line 29), there exists a problem with executing firmware update routines out of the firmware update. As the firmware update routines (from the incoming code image) are executing, the routines may attempt to call a task switching routine outside the scope of the update routines. This cannot be allowed to happen, as unpredictable results are likely to occur. The cited text from the DPA does not address how to solve this problem, but only states the limitations in the prior art method.

The Examiner cites col. 7, lines 27-28 of Gerard as teaching a method of updating a code image by modifying a method table pointer to point to the method table of another code image. The Examiner states that according to the cited passage, any method calls from a first image would then refer to a method body in the other image, thus preventing the execution of the method from the first image.

Gerard teaches an object identity swapper that dynamically updates the configuration of an object by taking a first object, instantiating a new second object, swapping the identities of the first and second objects, and reading and converting the state data of the old object (now the second object) into the new object (now the first object). The result is that the first object identity does not change, but its configuration is updated without passivating the object.

Gerard is only concerned with updating individual objects in an objected oriented environment while maintaining the individual objects' particular identities. The purpose for the method table pointers being swapped is to ensure that the object being updated does not have its identity changed. The object being updated is not executing.

In contrast, the update routine of Applicant's invention executes to accomplish an update of a current code image. A portion of the executing update routine (the task switching function) is prevented from executing because the executing update routine yields task switching control to the current code image. The method taught by Gerard has nothing to do with executing an update routine to update an entire code image while allowing the current code image to continue to execute during the update and preventing the executing update routine from initiating task switching functions.

Silberschatz teaches basic scheduling concepts and several different CPU algorithms. Multiprogramming and basic, simple process switching is discussed. Silberschatz does not teach or suggest retrieving, by the one or more code update routines, a task switching routine offset from the current code image resulting in the one or more code update routines transferring control to task switching functions of the current code image; and executing a task switching function from the current code image to switch microprocessor control from executing the one or more code update routines of the incoming image to execute a function in the current code image.

A solution to the previously described problem is taught in Applicant's amended independent claim 1. Support for the claim amendment can be found, at least, on page 14, lines 10-16 of the specification. Implicit in this description of the invention is the requirement that task switching only be initiated by the current code image and that any task switching functions in the incoming code image update routines be kept from executing. In other words, the function of the

update routines of the incoming code image is restricted to updating the firmware. If at anytime a task switching function is required, the update routines of incoming code image relinquish control to the current code image to execute task switching functions.

Independent claim 1 has been amended to clearly point out what the Applicant regards as the invention. Specifically, Applicant's amended independent claim 1 now recites, in part, (a) executing the current code image in the embedded system, (b) executing one or more code update routines from the incoming code image to update the current code image with the incoming code image, (c) retrieving, by the one or more code update routines, a task switching routine offset from the current code image resulting in the one or more code update routines transferring control to task switching functions of the current code image, and (d) executing a task switching function from the current code image to switch microprocessor control from executing the one or more code update routines of the incoming image to execute a function in the current code image.

The code update routines of the incoming image retrieve a task switching routine offset from the current code image so that task switching functions originate only from the current code image. This process prevents the update routines from initiating their own task switching routines.

In view of the foregoing, it is respectfully submitted that the DPA, Gerard, and Silberschatz, whether taken together or alone, do not teach or suggest the subject matter recited in Applicant's independent claim 1. Specifically, the cited references do not teach or suggest a method that involves retrieving, by the one or more code update routines, a task switching routine offset from the current code image resulting in the one or more code update routines transferring control to task switching functions of the current code image, and executing a task switching function from the current code image to switch microprocessor control from executing the one or more code update routines of the incoming image to execute a function in the current code image.

Claims 4, 6, and 8-11, which depend directly or indirectly from the independent claim 1, incorporate all of the limitations of the independent claim 1 and are therefore patentably distinct over the DPA, Gerard, and Silberschatz for at least those reasons provided for independent claim 1.

Independent claims 13 and 32 have also been amended to clearly point out the claimed invention and recite limitations similar to those recited in independent claim 1, and are therefore patentably distinct over the DPA, Gerard, and Silberschatz for at least those reasons provided for independent claim 1.

Claims 14, 16, 18-21, 32, 37, and 39-42, which depend directly or indirectly from the independent claims 13 and 32, incorporate all of the limitations of the corresponding independent claim and are therefore patentably distinct over the DPA, Gerard, and Silberschatz for at least those reasons provided for claims 13 and 32.

Rejection of Claims 2, 3, 28-30, 33, 34, and 44 under 35 U.S.C. §103(a)

The Office Action states that in an analogous environment, Levine teaches that the location of routines within a code segment can be represented by an offset from the beginning of the segment, and that it would have been obvious to use Levine's teaching of relocation offsets with the routines of DPA.

As previously discussed, the DPA, Gerard, and Silberschatz, whether taken alone or in combination, do not teach or suggest the subject matter recited in Applicant's independent claims 1, 13, 31, and 32. Specifically, the DPA, Gerard, and Silberschatz do not teach or suggest a method or system that involves retrieving, by the one or more code update routines, a task switching routine offset from the current code image resulting in the one or more code update routines transferring control to task switching functions of the current code image, and executing a task switching

function from the current code image to switch microprocessor control from executing the one or more code update routines of the incoming image to execute a function in the current code image.

Further, because the DPA, Gerard, and Silberschatz do not teach or suggest the subject matter recited in independent claims 1, 13, and 32, and because Levine does not teach or suggest the elements of claims 1, 13, and 32 that the DPA, Gerard, and Silberschatz are missing, Levine is irrelevant.

In view of the foregoing, it is respectfully submitted that DPA, Gerard, Silberschatz, and Levine, whether taken alone or in combination, do not teach or suggest the subject matter recited in claims 1, 13, and 32.

Claims 2, 28, 30, and 33, which depend directly or indirectly from the independent claims 1, 13, and 32, incorporate all of the limitations of the corresponding independent claim and are therefore patentably distinct over the DPA, Gerard, Silberschatz, and Levine for at least those reasons provided for claims 1, 13, and 32.

Independent claim 44 has been amended to recite limitations similar to those recited in independent claims 1, 13, and 32, and is therefore patentably distinct over the DPA, Gerard, Silberschatz, and Levine for at least those reasons provided for independent claims 1, 13, and 32.

Rejection of Claims 5, 15, 23, and 36 under 35 U.S.C. §103(a)

The Office Action states that in an analogous environment, Time-Life teaches that computers gather and distribute digital information using an input/output interface. Further, the Office Action states that Time-life teaches computer systems including a bus, microprocessor, RAM, programmable memory, and I/O interface.

As previously discussed, the DPA, Gerard, and Silberschatz, whether taken alone or in combination, do not teach or suggest the subject matter recited in Applicant's independent claims 1, 13, and 32. Specifically, the DPA, Gerard, and Silberschatz do not teach or suggest a method or system that involves retrieving, by the one or more code update routines, a task switching routine offset from the current code image resulting in the one or more code update routines transferring control to task switching functions of the current code image, and executing a task switching function from the current code image to switch microprocessor control from executing the one or more code update routines of the incoming image to execute a function in the current code image.

Further, because the DPA, Gerard, and Silberschatz do not teach or suggest the subject matter recited in independent claims 1, 13, and 32, and because Time-Life does not teach or suggest the elements of claims 1, 13, and 32 that the DPA, Gerard, and Silberschatz are missing, Time-Life is irrelevant.

In view of the foregoing, it is respectfully submitted that DPA, Gerard, Silberschatz, and Time-Life, whether taken alone or in combination, do not teach or suggest the subject matter recited in claims 1, 13, and 32.

Claims 5, 15, 23, and 36, which depend directly or indirectly from the independent claims 1, 13, and 32, incorporate all of the limitations of the corresponding independent claim and are therefore patentably distinct over the DPA, Gerard, Silberschatz, and Time-Life for at least those reasons provided for claims 1, 13, and 32.

Rejection of Claims 7, 17, and 38 under 35 U.S.C. §103(a)

The Office Action states that in an analogous environment, Amirghodsi teaches testing a pointer for a NULL value to determine validity.

As previously discussed, the DPA, Gerard, Silberschatz, and Levine, whether taken alone or in combination, do not teach or suggest the subject matter recited in Applicant's independent claims 1, 13, and 32. Specifically, the DPA, Gerard, Silberschatz, and Levine do not teach or suggest a method or system that involves retrieving, by the one or more code update routines, a task switching routine offset from the current code image resulting in the one or more code update routines transferring control to task switching functions of the current code image, and executing a task switching function from the current code image to switch microprocessor control from executing the one or more code update routines of the incoming image to execute a function in the current code image.

Further, as previously discussed, because the DPA, Gerard, Silberschatz, and Levine do not teach or suggest the subject matter recited in independent claims 1, 13, and 32, and because Amirghodsi does not teach or suggest the elements of claims 1, 13, and 32 that DPA, Gerard, Silberschatz, and Levine are missing, Amirghodsi is irrelevant.

In view of the foregoing, it is respectfully submitted that the DPA, Gerard, Silberschatz, Levine, and Amirghodsi, whether taken alone or in combination, do not teach or suggest the subject matter recited in claims 1, 13, and 32.

Claims 7, 17, and 38, which depend directly or indirectly from the independent claims 1, 13, and 32, incorporate all of the limitations of the corresponding independent claim and are therefore patentably distinct over the DPA, Gerard, Silberschatz, Levine, and Amirghodsi, for at least those reasons provided for claims 1, 13, and 32.

Rejection of Claim 24 under 35 U.S.C. §103(a)

The Office Action states that in an analogous environment, MCD further teaches that the process of integration combines multiple circuit elements on a single chip.

As previously discussed, the DPA, Gerard, Silberschatz, and Time-Life, whether taken alone or in combination, do not teach or suggest the subject matter recited in Applicant's independent claim 13. Specifically, the DPA, Gerard, Silberschatz, and Time-Life do not teach or suggest a method or system that involves retrieving, by the one or more code update routines, a task switching routine offset from the current code image resulting in the one or more code update routines transferring control to task switching functions of the current code image, and executing a task switching function from the current code image to switch microprocessor control from executing the one or more code update routines of the incoming image to execute a function in the current code image.

Further, as previously discussed, because the DPA, Gerard, Silberschatz, and Time-Life do not teach or suggest the subject matter recited in independent claim 13, and because MCD does not teach or suggest the elements of claim 13 that the DPA, Gerard, Silberschatz, and Time-Life are missing, MCD is irrelevant.

In view of the foregoing, it is respectfully submitted that DPA, Gerard, Silberschatz, Time-Life, and MCD, whether taken alone or in combination, do not teach or suggest the subject matter recited in claim 13 as each of these references fails at least to teach or suggest a method or system that involves retrieving, by the one or more code update routines, a task switching routine offset from the current code image resulting in the one or more code update routines transferring control to task switching functions of the current code image and executing a task switching function from

the current code image to switch microprocessor control from executing the one or more code update routines of the incoming image to execute a function in the current code image.

Claim 24, which depends directly or indirectly from the independent claim 13, incorporates all of the limitations of the independent claim 13 and is therefore patentably distinct over DPA, Gerard, Silberschatz, Time-Life, and MCD, for at least those reasons provided for claim 13.

Rejection of Claims 12 and 43 under 35 U.S.C. §103(a)

The Office Action states that in an analogous environment, Firooz teaches resetting a computer system upon completion of a code update.

As previously discussed, the DPA, Gerard, and Silberschatz, whether taken alone or in combination, do not teach or suggest the subject matter recited in Applicant's independent claims 1 and 32. Specifically, the DPA, Gerard, and Silberschatz do not teach or suggest a method or system that involves retrieving, by the one or more code update routines, a task switching routine offset from the current code image resulting in the one or more code update routines transferring control to task switching functions of the current code image and executing a task switching function from the current code image to switch microprocessor control from executing the one or more code update routines of the incoming image to execute a function in the current code image.

Further, because the DPA, Gerard, and Silberschatz do not teach or suggest the subject matter recited in independent claims 1 and 32, and because Firooz does not teach or suggest the elements of claims 1 and 32 that the DPA, Gerard, and Silberschatz are missing, Firooz is irrelevant.

In view of the foregoing, it is respectfully submitted that the DPA, Gerard, Silberschatz, and Firooz, whether taken alone or in combination, do not teach or suggest the subject matter recited in claims 1 and 32.

Claims 12 and 43, which depend directly or indirectly from the independent claims 1 and 32, incorporate all of the limitations of the corresponding independent claim and are therefore patentably distinct over the DPA, Gerard, Silberschatz and Firooz for at least those reasons provided for claims 1 and 32.

Rejection of Claims 22 and 25-27 under 35 U.S.C. §103(a)

The Office Action states that in an analogous environment, Samuels teaches bootloaders for instructing a processor to execute a code image.

As previously discussed, the DPA, Gerard, Silberschatz and Firooz, whether taken alone or in combination, do not teach or suggest the subject matter recited in Applicant's independent claim 13. Specifically, the DPA, Gerard, Silberschatz and Firooz do not teach or suggest a method or system that involves retrieving, by the one or more code update routines, a task switching routine offset from the current code image resulting in the one or more code update routines transferring control to task switching functions of the current code image and executing a task switching function from the current code image to switch microprocessor control from executing the one or more code update routines of the incoming image to execute a function in the current code image.

Further, as previously stated, because the DPA, Gerard, Silberschatz, and Firooz do not teach or suggest the subject matter recited in independent claim 13, and because Samuels does not teach or suggest the elements of claim 13 that the DPA, Gerard, Silberschatz, and Firooz are missing, Samuels is irrelevant.

In view of the foregoing, it is respectfully submitted that the DPA, Gerard, Silberschatz, Firooz, and Samuels, whether taken alone or in combination, do not teach or suggest the subject matter recited in claim 13.

Claims 22 and 25-27, which depend directly or indirectly from the independent claim 13, incorporate all of the limitations of the independent claim 13 and are therefore patentably distinct over the DPA, Gerard, Silberschatz, Firooz, and Samuels, for at least those reasons provided for claim 13.

Conclusion

In view of the foregoing, applicant respectfully requests reconsideration, withdrawal of all rejections, and allowance of all pending claims in due course.

Respectfully submitted,

Steven Fischman
Registration No. 34,594

SCULLY, SCOTT, MURPHY & PRESSER
400 Garden City Plaza
Suite 300
Garden City, New York 11530
(516) 742-4343

SF:BMM:ej

G:\IBM\1545\14942\AMEND\14942.AM2.doc